SCAN_IN *143* *102*
(scan input) *141*
LBRAM_SCAN_IN
(ramscan input)
LBRAM_SCAN_MODE
(ramscan mode) *134*
SCAN_ENABLE
(scan enable) *146*
aadr
(read address)
cpa
(read clock) *104*

LBRAM_SCAN_CLOCK
(scan clock) *136* *106*

cpb
(write clock)
web *148*
(write enable)
badr
(write address)

dib
(data input)
SCAN_MODE
(scan mode) *140*
110

tl

Read
Address
Registers *116*

te

d

te to

Lockup Latch *112*

te ti

Write
Enable
Register(s) *118*

d

d Write
Address
Registers *120*

te to

te ti

Input
Data
Registers *122*

s
0
1 to

126 *128*

Write
Decode
124

Read
Decode

Input
Data
Buffers

Latch
Array *130*

tm
(transparent
mode) *138*

0
1 *108*

Output
Data
Buffers *132*

doa
(output data)

LBRAM_SCAN_OUT_LD
(scan output)

114

Lockup Latch

SCAN_OUT_LD
(scan output)
142 *144*

FIG._1

FIG._2

FIG._3

```
Step 1:
for i = 0, i < r, i++
            if (i is even)
                        Wi(AAAA)
            else
                        Wi(5555)
            end
    end

Step 2:
R0(AAAA)
for i = 1, i < r, i++
                if (i is even)
                            Ri(AAAA)
                            Wi-1(AAAA)
                else
                            Ri(5555)
                            Wi-1(5555)
                end
    end
    if (r is even)
                Wr-1(AAAA)
    else
                Wr-1(5555)
    end

Step 3:
    if (r is even)
                Rr-1(AAAA)
    else
                Rr-1(5555)
    end
    for i = r-1, i > 0, i--
                if (i is even)
                            Ri-1(AAAA)
                            Wi(0000)
                else
                            Ri-1(5555)
                            Wi(FFFF)
                end
    end
    W0(0000)

Step 4:
DIB0 = FFFF
for i = 1, i < r, i++
                if (i is even)
                            DIBi = FFFF
                            Ri-1(FFFF)
                else
                            DIBi = 0000
                            Ri-1(0000)
                end
    end
    if (r is even)
                Rr-1(FFFF)
    else
                Rr-1(0000)
    end
```
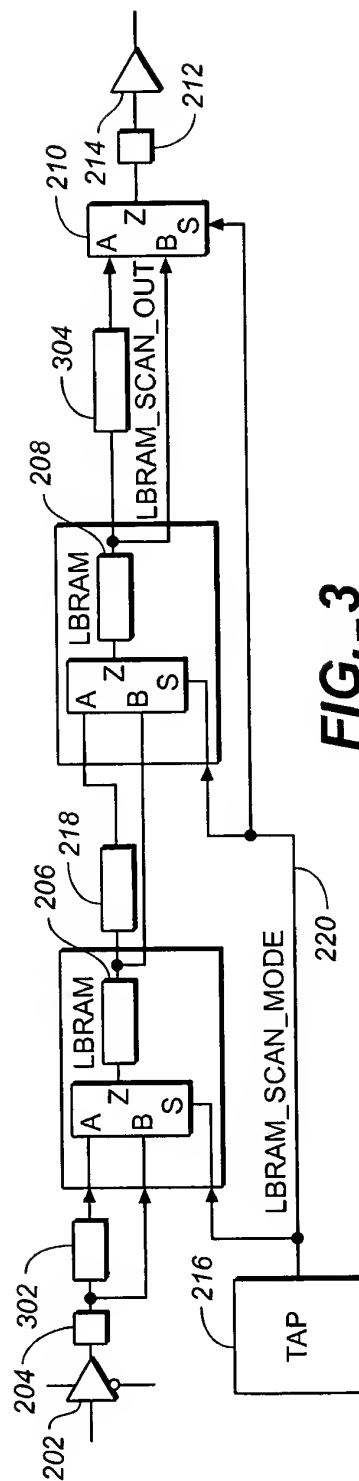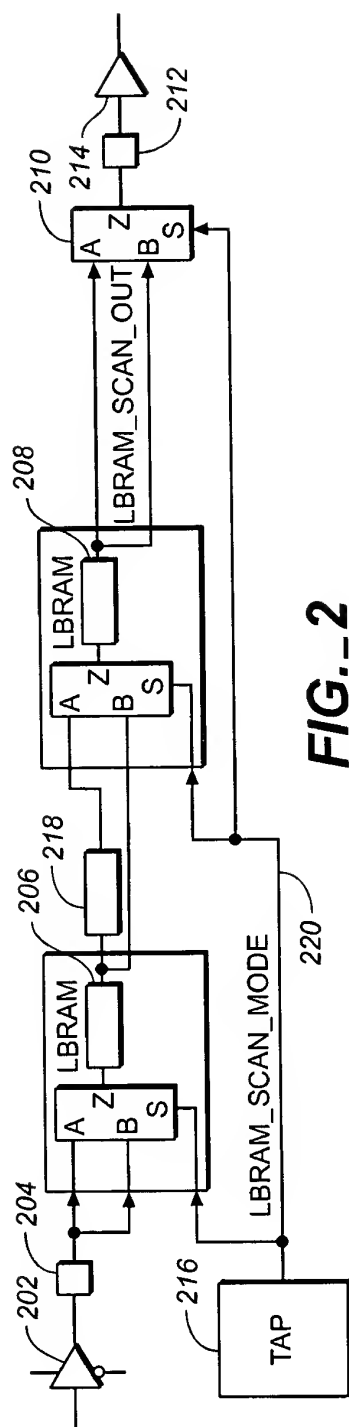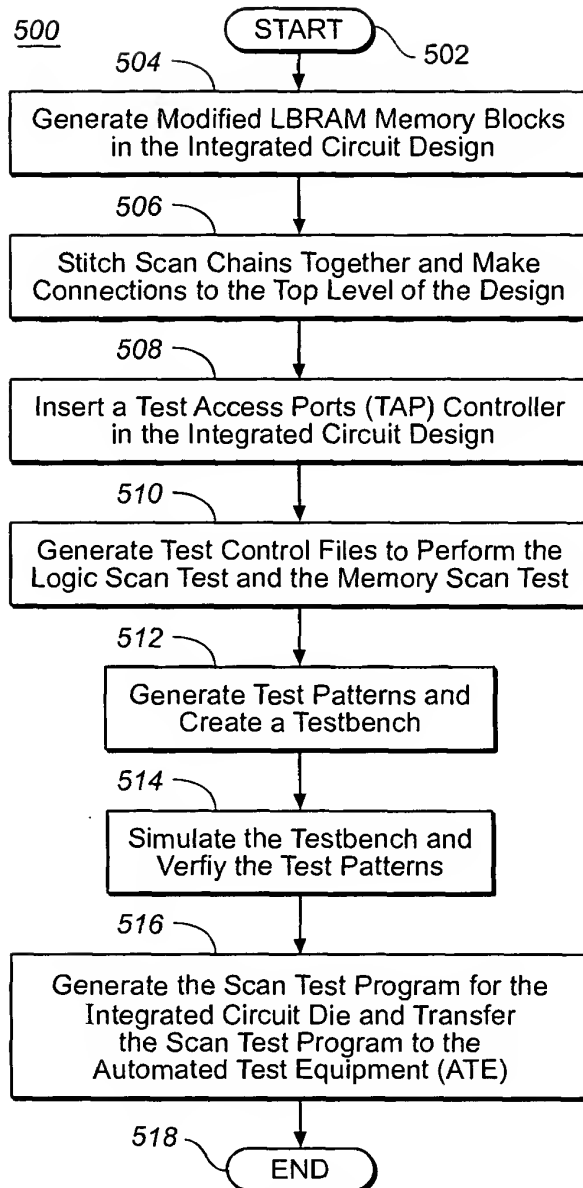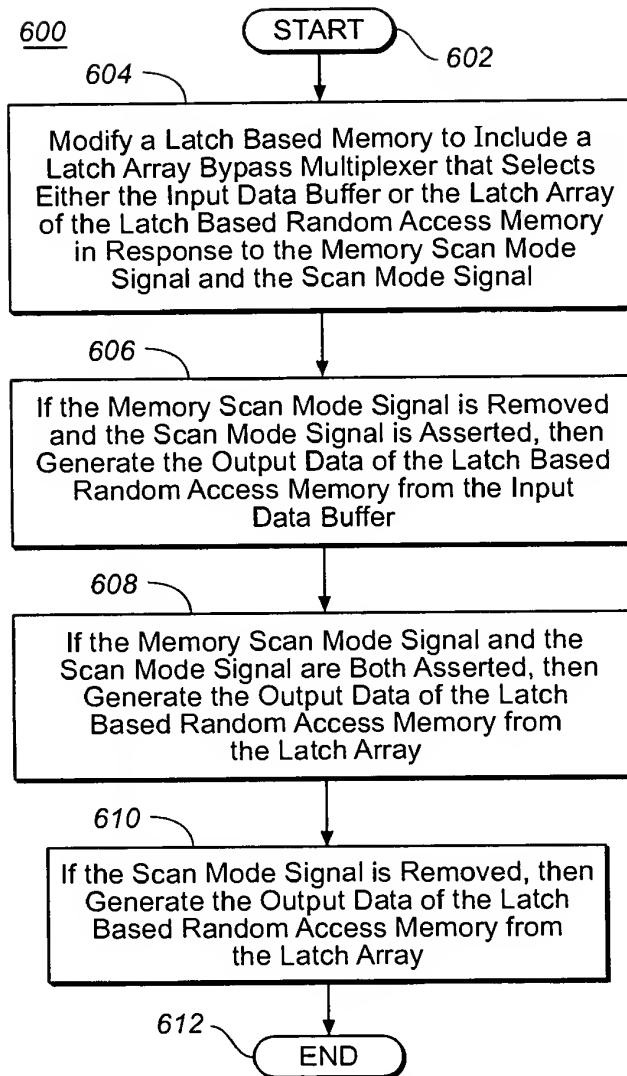
FIG._4

500

START

504 — *502*

Generate Modified LBRAM Memory Blocks
in the Integrated Circuit Design

506

Stitch Scan Chains Together and Make
Connections to the Top Level of the Design

508

Insert a Test Access Ports (TAP) Controller
in the Integrated Circuit Design

510

Generate Test Control Files to Perform the
Logic Scan Test and the Memory Scan Test

512

Generate Test Patterns and
Create a Testbench

514

Simulate the Testbench and
Verfiy the Test Patterns

516

Generate the Scan Test Program for the
Integrated Circuit Die and Transfer
the Scan Test Program to the
Automated Test Equipment (ATE)

518

END

FIG._5

5 / 5

600

START

602

604

Modify a Latch Based Memory to Include a
Latch Array Bypass Multiplexer that Selects
Either the Input Data Buffer or the Latch Array
of the Latch Based Random Access Memory
in Response to the Memory Scan Mode
Signal and the Scan Mode Signal

606

If the Memory Scan Mode Signal is Removed
and the Scan Mode Signal is Asserted, then
Generate the Output Data of the Latch Based
Random Access Memory from the Input
Data Buffer

608

If the Memory Scan Mode Signal and the
Scan Mode Signal are Both Asserted, then
Generate the Output Data of the Latch
Based Random Access Memory from
the Latch Array

610

If the Scan Mode Signal is Removed, then
Generate the Output Data of the Latch
Based Random Access Memory from
the Latch Array

612

END

FIG._6